Penalty-Based Multitask Distributed Adaptation over Networks with Constraints

Fei Hua, Roula Nassif, Cédric Richard, Haiyan Wang, Jianguo Huang

Penalty-Based Multitask Distributed Adaptation over Networks with Constraints

Fei Hua*†, Roula Nassif†, Cédric Richard†, Haiyan Wang*, Jianguo Huang*
*School of Marine Science and Technology
Northwestern Polytechnical University, Xi'an 710072, China
† Laboratoire Lagrange, Université Côte d'Azur, OCA, CNRS, Nice 06108, France
Email: {fei.hua, roula.nassif}@oca.eu cedric.richard@unice.fr {hywang, jghuang}@nwpu.edu.cn

***Abstract*—Multitask distributed optimization over networks enables the agents to cooperate locally to estimate multiple related parameter vectors. In this work, we consider multitask estimation problems over mean-square-error (MSE) networks where each agent is interested in estimating its own parameter vector, also called task, and where the tasks are related according to a set of linear equality constraints. We assume that each agent possesses its own cost and that the set of constraints is distributed among the agents. In order to solve the multitask problem, a cooperative algorithm based on penalty method is derived. Some results on its stability and convergence properties are also provided. Simulations are conducted to illustrate the theoretical results and show the efficiency of the strategy.**

I. Introduction

Distributed adaptive learning strategies over networks enable the agents to accomplish a certain task such as parameter estimation collaboratively from streaming data, and endow the agents with continuous adaptation and learning ability to track possible drifts in the underlying model. Although a centralized strategy may benefit more from information collected throughout the network, in most cases, distributed strategies are more attractive since they are scalable and robust. There is an extensive literature on distributed adaptive methods for single-task problems, where all the agents over the network have a common parameter vector to estimate [1]–[4]. However, many applications are multi-task oriented in the sense that the agents have to infer multiple parameter vectors simultaneously. In this case, the agents do not share a common minimizer. It is shown in [5] that the network converges to a Pareto solution corresponding to a multi-objective optimization problem. Multitask diffusion strategies, by exploiting prior information about relationships between the tasks, can let the agents or clusters of agents converge to their own respective models. One useful way to model relationships among tasks is to formulate optimization problems with appropriate regularizers between agents [6]–[9]. In [10]–[12], distributed algorithms are derived to estimate node-specific parameter vectors that lie in a common latent signal subspace. In other works [13], [14], the parameter space is decomposed into two orthogonal subspaces. The relations among tasks are modeled by assuming that they all share one of the subspaces. In [15], it is assumed that each agent has only access to a subset of the entries of a global parameter vector and only shares the common entries with its neighbors.

In some applications, it happens that each agent has its own parameter vector to estimate and these vectors are coupled together through a set of linear constraints. Examples include the network flow control problem [16], the interference management problem in communication networks [17], and the basis pursuit problem [18]. In this work, we consider multitask estimation problems where the parameter vectors to be estimated at neighboring agents are related according to a set of linear equality constraints. Therefore, the objective of the network is to optimize the aggregate cost across all nodes subject to all constraints:

$$\underset{\boldsymbol{w}_1,\ldots,\boldsymbol{w}_N}{\text{minimize}} \quad J^{\text{glob}}(\boldsymbol{w}_1,\ldots,\boldsymbol{w}_N) \triangleq \sum_{k=1}^{N} J_k(\boldsymbol{w}_k), \tag{1a}$$

$$\text{subject to} \quad \sum_{\ell\in\mathcal{I}_p} \boldsymbol{D}_{p\ell}\boldsymbol{w}_\ell + \boldsymbol{b}_p = 0, \quad p=1,\ldots,P \tag{1b}$$

with N the number of agents in the network. Each agent k seeks to estimate its own parameter vector $\boldsymbol{w}_k \in \mathbb{R}^{M_k\times 1}$, and has knowledge of its local cost $J_k(\cdot)$ and the set of linear equality constraints that it is involved in. The dimension of the parameter vectors can differ from one node to another. Each constraint is indexed by p, and defined by the $L_p \times M_\ell$ matrix $\boldsymbol{D}_{p\ell}$, the $L_p \times 1$ vector $\boldsymbol{b}_p$, and the set $\mathcal{I}_p$ of agent indices involved in the p-th constraint. It is assumed that each agent k in $\mathcal{I}_p$ can collect information from the other agents in $\mathcal{I}_p$, i.e., $\mathcal{I}_p \subseteq \mathcal{N}_k$ for all $k \in \mathcal{I}_p$ where $\mathcal{N}_k$ denotes the neighborhood of agent k.

In a previous work [19], two of the authors of this paper address (1) by combining diffusion adaptation with a stochastic projection method. The nodes involved in several constraints are divided into virtual sub-nodes in order to circumvent the problem of projecting their local parameter vector onto several constraint subspaces simultaneously. In this work, we propose an alternative method that consists of reformulating (1) as an unconstrained problem with penalty

The work of F. Hua was supported in part by China Scholarship Council and NSFC grant 61471298. The work of R. Nassif and C. Richard was partly supported by ANR and DGA grant ANR-13-ASTR-0300 (ODISSEE project). The work of H. Wang was partly supported by NSFC under grants 61571365 and 61671386.

functions. We devise a distributed learning strategy relying on an adaptation step and a penalization step. Although we consider only the case of equality constraints, the algorithm can be easily extended to solve problems with inequality constraints. We analyze its behavior in the mean and mean-square-error sense. Simulations are conducted to show the effectiveness of the proposed strategy.

Notation: The symbol $\otimes_b$ denotes the block Kronecker product, and the symbol bvec$(\cdot)$ refers to the block vectorization operation that vectorizes each block of its matrix argument and stacks the vectors on top of each other.

II. Problem formulation and distributed solution

Consider a network of N agents, labeled with $k = 1, \ldots, N$. At each time instant $i \geq 0$, each agent k is assumed to have access to a zero-mean scalar measurement $d_k(i)$ and a real-valued regression vector $\boldsymbol{x}_k(i) \in \mathbb{R}^{M_k \times 1}$ with positive covariance matrix $\boldsymbol{R}_{x,k} = \mathbb{E}\{\boldsymbol{x}_k(i)\boldsymbol{x}_k^\top(i)\}$. The data $\{d_k(i), \boldsymbol{x}_k(i)\}$ are assumed to be related via the linear regression model:

$$d_k(i) = \boldsymbol{x}_k^\top(i)\boldsymbol{w}_k^o + z_k(i) \tag{2}$$

where $\boldsymbol{w}_k^o$ is an unknown parameter vector, and $z_k(i)$ is a zero-mean measurement noise with variance $\sigma_{z,k}^2$ assumed to be spatially and temporally independent.

Let $\boldsymbol{w}_k \in \mathbb{R}^{M_k \times 1}$ denote the parameter vector associated with agent k. The objective at agent k is to estimate $\boldsymbol{w}_k^o$ by minimizing the cost function $J_k(\boldsymbol{w}_k)$ given by:

$$J_k(\boldsymbol{w}_k) = \mathbb{E}|d_k(i) - \boldsymbol{x}_k^\top(i)\boldsymbol{w}_k|^2 \tag{3}$$

We assume that $J_k(\cdot)$ is strongly convex and second-order differentiable. In addition, we consider that the optimum parameter vectors at neighboring agents are related according to a set of linear equality constraints of the form (1b). Each agent k has knowledge of its cost and the set of constraints that it is involved in. We use $\mathcal{J}_k$ to denote the set of constraint indices involving agent k, i.e., $\mathcal{J}_k \triangleq \{p | k \in \mathcal{I}_p\}$.

We collect the parameter vectors $\boldsymbol{w}_k$ and $\boldsymbol{w}_k^o$ from across all nodes into the following $N \times 1$ block vectors:

$$\boldsymbol{w} \triangleq \text{col}\{\boldsymbol{w}_1, \ldots, \boldsymbol{w}_N\}, \quad \boldsymbol{w}^o \triangleq \text{col}\{\boldsymbol{w}_1^o, \ldots, \boldsymbol{w}_N^o\}. \tag{4}$$

The constraints in (1b) can be written more compactly as:

$$\boldsymbol{\mathcal{D}}\boldsymbol{w} + \boldsymbol{b} = \boldsymbol{0} \tag{5}$$

where $\boldsymbol{\mathcal{D}}$ is a $P \times N$ block matrix, with each block $\boldsymbol{D}_{p\ell}$ having dimension $L_p \times M_\ell$, and $\boldsymbol{b}$ is a $P \times 1$ block column vector with each block $\boldsymbol{b}_p$ having dimensions $L_p \times 1$. We assume $\boldsymbol{\mathcal{D}}$ is full row rank to ensure that (5) has at least one solution. This leads to the network constrained optimization problem:

$$\begin{aligned} \underset{\boldsymbol{w}}{\text{minimize}} \quad & \sum_{k=1}^{N} \mathbb{E}|d_k(i) - \boldsymbol{x}_k^\top(i)\boldsymbol{w}_k|^2 \\ \text{subject to} \quad & \boldsymbol{\mathcal{D}}\boldsymbol{w} + \boldsymbol{b} = \boldsymbol{0}. \end{aligned} \tag{6}$$

Let $\boldsymbol{r}_{dx,k} \triangleq \mathbb{E}\{d_k(i)\boldsymbol{x}_k(i)\}$ and $\sigma_{d,k}^2 \triangleq \mathbb{E}|d_k(i)|^2$. Problem (6) can be written equivalently as:

$$\begin{aligned} \underset{\boldsymbol{w}}{\text{minimize}} \quad & \boldsymbol{w}^\top \boldsymbol{\mathcal{R}}_x \boldsymbol{w} - 2\boldsymbol{r}_{dx}^\top \boldsymbol{w} + \boldsymbol{\sigma}_d^\top \mathbb{1}_{N\times 1}, \\ \text{subject to} \quad & \boldsymbol{\mathcal{D}}\boldsymbol{w} + \boldsymbol{b} = \boldsymbol{0} \end{aligned} \tag{7}$$

where the $N \times N$ block diagonal matrix $\boldsymbol{\mathcal{R}}_x$, the $N \times 1$ block vector $\boldsymbol{r}_{dx}$, and the $N \times 1$ vector $\boldsymbol{\sigma}_d$ are given by:

$$\boldsymbol{\mathcal{R}}_x \triangleq \text{diag}\{\boldsymbol{R}_{x,1}, \ldots, \boldsymbol{R}_{x,N}\}, \tag{8}$$

$$\boldsymbol{r}_{dx} \triangleq \text{col}\{\boldsymbol{r}_{dx,1}, \ldots, \boldsymbol{r}_{dx,2}\}, \tag{9}$$

$$\boldsymbol{\sigma}_d \triangleq \text{col}\{\sigma_{d,1}^2, \ldots, \sigma_{d,N}^2\}. \tag{10}$$

Because $\boldsymbol{\mathcal{R}}_x$ is positive definite, problem (7) is a positive definite quadratic program with equality constraints. It has a unique global minimum given by:

$$\boldsymbol{w}^\star = \boldsymbol{w}^o - \boldsymbol{\mathcal{R}}_x^{-1}\boldsymbol{\mathcal{D}}^\top(\boldsymbol{\mathcal{D}}\boldsymbol{\mathcal{R}}_x^{-1}\boldsymbol{\mathcal{D}}^\top)^{-1}(\boldsymbol{\mathcal{D}}\boldsymbol{w}^o + \boldsymbol{b}). \tag{11}$$

By augmenting the objective function with a penalty term, problem (1) can be approximated into an unconstrained problem of the following form:

$$\underset{\boldsymbol{w}_1, \ldots, \boldsymbol{w}_N}{\text{minimize}} \quad \sum_{k=1}^{N} J_k(\boldsymbol{w}_k) + \eta \sum_{p=1}^{P} \| \sum_{\ell \in \mathcal{I}_p} \boldsymbol{D}_{p\ell}\boldsymbol{w}_\ell + \boldsymbol{b}_p \|^2 \tag{12}$$

where $\eta > 0$ is a scalar parameter that controls the relative importance of adhering to the constraints. The approximation (12) of (1) improves in quality as η increases [20], [21]. Problem (12) can be written alternatively as:

$$\underset{\boldsymbol{w}}{\text{minimize}} \quad J_\eta^{\text{glob}}(\boldsymbol{w}) \triangleq J^{\text{glob}}(\boldsymbol{w}) + \eta \|\boldsymbol{\mathcal{D}}\boldsymbol{w} + \boldsymbol{b}\|^2 \tag{13}$$

where $J^{\text{glob}}(\cdot)$ is given in (1a). The above problem is strongly convex for any η and its closed form solution parameterized by η is given by:

$$\boldsymbol{w}^o(\eta) = (\boldsymbol{\mathcal{R}}_x + \eta\boldsymbol{\mathcal{D}}^\top\boldsymbol{\mathcal{D}})^{-1}(\boldsymbol{\mathcal{R}}_x\boldsymbol{w}^o - \eta\boldsymbol{\mathcal{D}}^\top\boldsymbol{b}). \tag{14}$$

Node k can apply a steepest-descent iteration to minimize the cost in (12) with respect to $\boldsymbol{w}_k$. Starting from an initial condition $\boldsymbol{w}_k(0)$, we obtain the following steepest descent iteration at node k:

$$\begin{aligned} \boldsymbol{w}_k(i+1) = \boldsymbol{w}_k(i) - \mu\Big[& \boldsymbol{R}_{x,k}\boldsymbol{w}_k(i) - \boldsymbol{r}_{dx,k} + \\ & \eta \sum_{p \in \mathcal{J}_k} \boldsymbol{D}_{pk}^\top \Big(\sum_{\ell \in \mathcal{I}_p \subseteq \mathcal{N}_k} \boldsymbol{D}_{p\ell}\boldsymbol{w}_\ell(i) + \boldsymbol{b}_p \Big) \Big]. \end{aligned} \tag{15}$$

Replacing the second-order moments $\boldsymbol{R}_{x,k}$, $\boldsymbol{r}_{dx,k}$ by instantaneous approximations:

$$\boldsymbol{R}_{x,k} \approx \boldsymbol{x}_k(i)\boldsymbol{x}_k^\top(i), \quad \boldsymbol{r}_{dx,k} \approx d_k(i)\boldsymbol{x}_k(i) \tag{16}$$

and implementing the update iteration into two successive steps by introducing the intermediate estimate $\boldsymbol{\phi}_k(i+1)$, we obtain the following adaptive algorithm at agent k:

$$\boldsymbol{\phi}_k(i+1) = \boldsymbol{w}_k(i) + \mu\boldsymbol{x}_k(i)\Big(d_k(i) - \boldsymbol{x}_k^\top(i)\boldsymbol{w}_k(i)\Big), \tag{17a}$$

$$\begin{aligned} \boldsymbol{w}_k(i+1) = {} & \boldsymbol{\phi}_k(i+1) \\ & - \mu\eta \sum_{p \in \mathcal{J}_k} \boldsymbol{D}_{pk}^\top \Big(\sum_{\ell \in \mathcal{I}_p \subseteq \mathcal{N}_k} \boldsymbol{D}_{p\ell}\boldsymbol{\phi}_\ell(i+1) + \boldsymbol{b}_p \Big). \end{aligned} \tag{17b}$$

Note that in the second step (17b), $\boldsymbol{w}_\ell(i)$ is replaced by the intermediate estimate $\boldsymbol{\phi}_\ell(i+1)$ which is a better estimate for the solution at agent ℓ. In the first step (17a), which corresponds to the adaptation step, node k uses its data to update its estimate $\boldsymbol{w}_k(i)$ to an intermediate estimate $\boldsymbol{\phi}_k(i+1)$. In the second step (17b), which is the penalization step, node k collects the intermediate estimates $\boldsymbol{\phi}_\ell(i+1)$ from its neighbors and moves along the gradient of the penalty function. Note that, in this step, instead of sending $\boldsymbol{\phi}_\ell(i+1)$ to agent k, agent ℓ may send the vector $\boldsymbol{D}_{p\ell}\boldsymbol{\phi}_\ell(i+1)$. In this sense, our algorithm has privacy-preserving property.

III. Performance Analysis

We shall now analyze the mean and mean-square-error behavior of the adaptive algorithm (17) with respect to $\boldsymbol{w}^o$, $\boldsymbol{w}^o(\eta)$, and $\boldsymbol{w}^\star$. Due to space limitations, detailed proofs or derivations are omitted. Before proceeding, let us introduce the following assumption.

Assumption 1. *The regressors $\boldsymbol{x}_k(i)$ arise from a zero-mean random process that is temporally white and spatially independent.*

Under this assumption, $\boldsymbol{x}_k(i)$ is independent of $\boldsymbol{w}_\ell(j)$ for all $i \geq j$ and for all ℓ. This assumption is commonly used in the adaptive filtering literature because it helps to simplify the analysis without constraining the conclusions [22].

A. Mean error behavior analysis

Let us introduce the error vectors $\widetilde{\boldsymbol{w}}_k(i) \triangleq \boldsymbol{w}_k^o - \boldsymbol{w}_k(i)$ and collect them into the network block error vector:

$$\widetilde{\boldsymbol{w}}(i) = \text{col}\{\widetilde{\boldsymbol{w}}_1(i), \ldots, \widetilde{\boldsymbol{w}}_N(i)\}. \tag{18}$$

We also introduce the following notations:

$$\boldsymbol{\mathcal{H}} \triangleq [\boldsymbol{I}_M - \mu\eta\boldsymbol{\mathcal{D}}^\top\boldsymbol{\mathcal{D}}], \tag{19}$$

$$\boldsymbol{\mathcal{B}} \triangleq \boldsymbol{\mathcal{H}}[\boldsymbol{I}_M - \mu\boldsymbol{\mathcal{R}}_x], \tag{20}$$

$$\boldsymbol{f}_o \triangleq \boldsymbol{\mathcal{D}}^\top(\boldsymbol{\mathcal{D}}\boldsymbol{w}^o + \boldsymbol{b}), \tag{21}$$

where $M \triangleq \sum_{k=1}^N M_k$. It can be verified that the network mean error vector $\mathbb{E}\,\widetilde{\boldsymbol{w}}(i)$ evolves according to:

$$\mathbb{E}\widetilde{\boldsymbol{w}}(i+1) = \boldsymbol{\mathcal{B}}\mathbb{E}\widetilde{\boldsymbol{w}}(i) + \mu\eta\boldsymbol{f}_o. \tag{22}$$

Recursion (22) converges as $i \to \infty$ if the matrix $\boldsymbol{\mathcal{B}}$ is stable. The stability of $\boldsymbol{\mathcal{B}}$ is ensured by choosing μ such that:

$$0 < \mu < \min\Big\{\frac{2}{\lambda_{\max}(\boldsymbol{R}_{x,k})}, \frac{2}{\eta\cdot\lambda_{\max}(\boldsymbol{\mathcal{D}}^\top\boldsymbol{\mathcal{D}})}\Big\},\ k = 1, \ldots, N. \tag{23}$$

In this case, the asymptotic mean bias is given by:

$$\mathbb{E}\widetilde{\boldsymbol{w}}(\infty) = \lim_{i\to\infty}\mathbb{E}\widetilde{\boldsymbol{w}}(i) = \mu\eta(\boldsymbol{I}_M - \boldsymbol{\mathcal{B}})^{-1}\boldsymbol{f}_o. \tag{24}$$

Using similar arguments, we can derive the mean error recursion with respect to $\boldsymbol{w}^o(\eta)$ and $\boldsymbol{w}^\star$. Let $\boldsymbol{w}_\eta^\delta \triangleq \boldsymbol{w}^o(\eta) - \boldsymbol{w}^o$, $\boldsymbol{w}_\star^\delta \triangleq \boldsymbol{w}^\star - \boldsymbol{w}^o$. Let us introduce the network error vectors $\widetilde{\boldsymbol{w}}'(i) = \boldsymbol{w}^o(\eta) - \boldsymbol{w}(i)$ and $\widetilde{\boldsymbol{w}}''(i) = \boldsymbol{w}^\star - \boldsymbol{w}(i)$. The mean error recursions with respect to $\boldsymbol{w}^o(\eta)$ and $\boldsymbol{w}^\star$ are given by:

$$\mathbb{E}\widetilde{\boldsymbol{w}}'(i+1) = \boldsymbol{\mathcal{B}}\mathbb{E}\widetilde{\boldsymbol{w}}'(i) + \mu(\boldsymbol{r}_e + \eta\boldsymbol{f}_e), \tag{25}$$

$$\mathbb{E}\widetilde{\boldsymbol{w}}''(i+1) = \boldsymbol{\mathcal{B}}\mathbb{E}\widetilde{\boldsymbol{w}}''(i) + \mu\boldsymbol{r}_s, \tag{26}$$

where

$$\boldsymbol{r}_e \triangleq \boldsymbol{\mathcal{H}}\boldsymbol{\mathcal{R}}_x\boldsymbol{w}_\eta^\delta,$$
$$\boldsymbol{r}_s \triangleq \boldsymbol{\mathcal{H}}\boldsymbol{\mathcal{R}}_x\boldsymbol{w}_\star^\delta,$$
$$\boldsymbol{f}_e \triangleq \boldsymbol{\mathcal{D}}^\top(\boldsymbol{\mathcal{D}}\boldsymbol{w}^o(\eta) + \boldsymbol{b}).$$

Under condition (23), the asymptotic biases are given by:

$$\mathbb{E}\widetilde{\boldsymbol{w}}'(\infty) = \mu(\boldsymbol{I}_M - \boldsymbol{\mathcal{B}})^{-1}(\boldsymbol{r}_e + \eta\boldsymbol{f}_e), \tag{27}$$

$$\mathbb{E}\widetilde{\boldsymbol{w}}''(\infty) = \mu(\boldsymbol{I}_M - \boldsymbol{\mathcal{B}})^{-1}\boldsymbol{r}_s. \tag{28}$$

Observe that when $\boldsymbol{w}^o$ satisfies the linear constraints, i.e., $\boldsymbol{w}^o = \boldsymbol{w}^o(\eta) = \boldsymbol{w}^\star$, all the biases reduce to zero.

B. Mean-square-error behavior analysis

To analyze the mean-square-error stability, we evaluate the weighted mean-square deviation weighted by any positive semi-definite matrix $\boldsymbol{\Sigma}$. The freedom in selecting $\boldsymbol{\Sigma}$ allows us to extract various types of information about the network. It is convenient to introduce the alternative notation $\|\boldsymbol{w}\|_{\boldsymbol{\sigma}}^2$ to refer to the weighted square quantity $\|\boldsymbol{w}\|_{\boldsymbol{\Sigma}}^2$, where $\boldsymbol{\sigma} \triangleq \text{bvec}(\boldsymbol{\Sigma})$. We shall use these two notations interchangeably. It can be verified that $\mathbb{E}\{\|\widetilde{\boldsymbol{w}}(i+1)\|_{\boldsymbol{\sigma}}^2\}$ evolves according to:

$$\mathbb{E}\{\|\widetilde{\boldsymbol{w}}(i+1)\|_{\boldsymbol{\sigma}}^2\} = \mathbb{E}\{\|\widetilde{\boldsymbol{w}}(i)\|_{\boldsymbol{\mathcal{F}}\boldsymbol{\sigma}}^2\} + [\text{bvec}(\boldsymbol{\mathcal{Y}}(i))]^\top\boldsymbol{\sigma}, \tag{29}$$

where $\boldsymbol{\mathcal{F}}$ is the $M^2 \times M^2$ matrix given by:

$$\boldsymbol{\mathcal{F}} \triangleq \mathbb{E}\{\boldsymbol{\mathcal{B}}^\top(i) \otimes_b \boldsymbol{\mathcal{B}}^\top(i)\}. \tag{30}$$

The matrix $\boldsymbol{\mathcal{Y}}(i)$ is given by:

$$\boldsymbol{\mathcal{Y}}(i) \triangleq \mu^2\boldsymbol{\mathcal{G}}^\top + \mu^2\eta^2\boldsymbol{f}_o\boldsymbol{f}_o^\top + 2\mu\eta\boldsymbol{f}_0\mathbb{E}\widetilde{\boldsymbol{w}}^\top(i)\boldsymbol{\mathcal{B}}^\top, \tag{31}$$

where $\boldsymbol{\mathcal{G}}$ is the $M \times M$ matrix given by:

$$\boldsymbol{\mathcal{G}} \triangleq \boldsymbol{\mathcal{H}}\text{diag}\{\sigma_{z,k}^2\boldsymbol{R}_{x,k}\}_{k=1}^N\boldsymbol{\mathcal{H}}^\top. \tag{32}$$

The algorithm is mean-square stable if $\boldsymbol{\mathcal{F}}$ is stable[1]. For sufficiently small-step size, neglecting the influence of high-order terms in μ, $\boldsymbol{\mathcal{F}}$ can be approximated by $\boldsymbol{\mathcal{F}} \approx \boldsymbol{\mathcal{B}}^\top \otimes_b \boldsymbol{\mathcal{B}}^\top$ [1], [2] and condition (23) ensures mean-square stability. It can be verified that $\mathbb{E}\{\|\widetilde{\boldsymbol{w}}(i+1)\|_{\boldsymbol{\sigma}}^2\}$ evolves according to the following recursion:

$$\begin{aligned}\mathbb{E}\{\|\widetilde{\boldsymbol{w}}(i+1)\|_{\boldsymbol{\sigma}}^2\} &= \mathbb{E}\{\|\widetilde{\boldsymbol{w}}(i)\|_{\boldsymbol{\sigma}}^2\} + \\ &\big[\text{bvec}(\mathbb{E}\{\widetilde{\boldsymbol{w}}(0)\widetilde{\boldsymbol{w}}^\top(0)\})\big]^\top(\boldsymbol{\mathcal{F}} - I_{M^2})\boldsymbol{\mathcal{F}}^i\boldsymbol{\sigma} + \\ &[\text{bvec}(\boldsymbol{\mathcal{Y}}(i))]^\top\boldsymbol{\sigma} + \boldsymbol{\Gamma}(i)\boldsymbol{\sigma}\end{aligned} \tag{33}$$

where $\widetilde{\boldsymbol{w}}(0)$ is the initial condition and $\boldsymbol{\Gamma}(i+1)$ is a $1 \times M^2$ vector that can be evaluated from $\boldsymbol{\Gamma}(i)$ according to:

$$\boldsymbol{\Gamma}(i+1) = [\text{bvec}(\boldsymbol{\mathcal{Y}}(i))]^\top(\boldsymbol{\mathcal{F}} - \boldsymbol{I}_{M^2}) + \boldsymbol{\Gamma}(i) \tag{34}$$

[1]Note that, in the case of zero-mean Gaussian regressors, the matrix $\boldsymbol{\mathcal{F}}$ can be calculated in closed form, see [19, Appendix B].

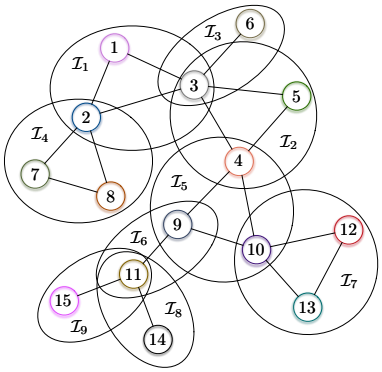

Fig. 1: Multitask MSE network with local constraints.

with $\boldsymbol{\Gamma}(0) = \mathbf{0}_{M^2}$.

The steady-state network MSD is given by:

$$\zeta^\star = \lim_{i\to\infty} \frac{1}{N}\mathbb{E}\{\|\widetilde{\boldsymbol{w}}(i)\|^2\}. \tag{35}$$

If the matrix $\boldsymbol{\mathcal{F}}$ is stable, from the recursion (29), we obtain as $i \to \infty$:

$$\lim_{i\to\infty} \mathbb{E}\{\|\widetilde{\boldsymbol{w}}(i)\|^2_{(\boldsymbol{I}_{M^2}-\boldsymbol{\mathcal{F}})\boldsymbol{\sigma}}\} = [\text{bvec}(\boldsymbol{\mathcal{Y}}(\infty))]^\top \boldsymbol{\sigma}. \tag{36}$$

Replacing $\boldsymbol{\sigma}$ in (36) by $\frac{1}{N}(\boldsymbol{I}_{M^2} - \boldsymbol{\mathcal{F}})^{-1}\text{bvec}(\boldsymbol{I}_M)$ we obtain:

$$\zeta^\star = \frac{1}{N}[\text{bvec}(\boldsymbol{\mathcal{Y}}(\infty))]^\top (\boldsymbol{I}_{M^2} - \boldsymbol{\mathcal{F}})^{-1}\text{bvec}(\boldsymbol{I}_M) \tag{37}$$

where $\boldsymbol{\mathcal{Y}}(\infty)$ can be obtained from (24) and (31).

The transient and steady-state behaviors of $\mathbb{E}\{\|\widetilde{\boldsymbol{w}}'(i)\|^2_{\boldsymbol{\sigma}}\}$ and $\mathbb{E}\{\|\widetilde{\boldsymbol{w}}''(i)\|^2_{\boldsymbol{\sigma}}\}$ can be derived from the model for $\widetilde{\boldsymbol{w}}(i)$ according to:

$$\mathbb{E}\{\|\widetilde{\boldsymbol{w}}'(i)\|^2_{\boldsymbol{\sigma}}\} = \mathbb{E}\{\|\widetilde{\boldsymbol{w}}(i)\|^2_{\boldsymbol{\sigma}}\} + 2\mathbb{E}\{\widetilde{\boldsymbol{w}}(i)\}\boldsymbol{\Sigma}\boldsymbol{w}_\eta^\delta + \|\boldsymbol{w}_\eta^\delta\|^2_{\boldsymbol{\Sigma}},$$
$$\mathbb{E}\{\|\widetilde{\boldsymbol{w}}''(i)\|^2_{\boldsymbol{\sigma}}\} = \mathbb{E}\{\|\widetilde{\boldsymbol{w}}(i)\|^2_{\boldsymbol{\sigma}}\} + 2\mathbb{E}\{\widetilde{\boldsymbol{w}}(i)\}\boldsymbol{\Sigma}\boldsymbol{w}_\star^\delta + \|\boldsymbol{w}_\star^\delta\|^2_{\boldsymbol{\Sigma}}.$$

IV. Simulations

We shall now provide simulation examples to illustrate the behavior of algorithm (17). We considered a network consisting of 15 nodes with connections shown in Fig. 1. The regression vectors $\boldsymbol{x}_k(i)$ were zero-mean Gaussian with covariance matrix $\boldsymbol{R}_{x,k} = \sigma^2_{x,k}\boldsymbol{I}_2$. The noises $z_k(i)$ were zero-mean i.i.d. Gaussian random variables independent of any other signal with variances $\sigma^2_{z,k}$. The variances $\sigma^2_{x,k}$ and $\sigma^2_{z,k}$ are shown in Fig. 2. We randomly sampled 9 linear constraints of the form $\sum_{\ell\in\mathcal{I}_p} d_{p\ell}\boldsymbol{w}_\ell = b_p \cdot \mathbb{1}_{2\times 1}$, where the coefficients $d_{p\ell}$ and b_p were randomly chosen from $\{-2,-1,1,2\}$. The results were averaged over 200 Monte-Carlo runs.

In the first scenario, we considered the case of a perfect model where the parameter vector $\boldsymbol{w}^o$ satisfies the constraints, i.e. $\boldsymbol{w}^o = \boldsymbol{w}^\star$. The step size μ was set to 0.02 for all nodes. In Fig. 3, we compare three algorithms: the non-cooperative LMS algorithm, the centralized CLMS algorithm [23], and the proposed algorithm (17). We observe that the simulation results match well the actual performance. Furthermore, compared to the non-cooperative strategy, the network MSD is

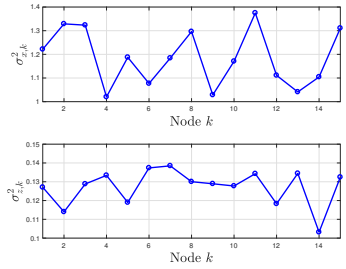

Fig. 2: Regression and noise variances.

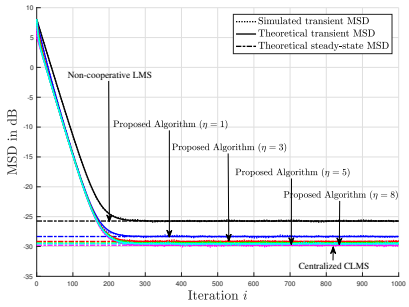

Fig. 3: MSD comparison of different algorithms for the perfect model scenario.

improved by the penalty term which, in this case, promotes the relationship between tasks. Finally, our algorithm performs well compared to the centralized solution and the gap w.r.t. centralized performance decreases as η increases.

In a second scenario, we considered the case when $\boldsymbol{w}^o$ does not satisfy the constraints. We perturbed $\boldsymbol{w}^o$ as $\boldsymbol{w}_{\text{pert}} = \boldsymbol{w}^o + \boldsymbol{u}$. The entries of $\boldsymbol{u}$ were sampled from Gaussian distribution $\mathcal{N}(0,\sigma^2)$. We set $\mu = 0.02$ and $\eta = 8$. The theoretical and simulated learning curves with respect to $\boldsymbol{w}^o$, $\boldsymbol{w}^o(\eta)$, and $\boldsymbol{w}^\star$ are depicted in Fig. 4. Although the performance with respect to $\boldsymbol{w}^o$ deteriorates as σ increases, the algorithm performs well with respect to $\boldsymbol{w}^o(\eta)$ and $\boldsymbol{w}^\star$.

Next, we illustrate in Fig. 5 *(left)* the MSD curves of the centralized algorithm and the proposed algorithm w.r.t. $\boldsymbol{w}^\star$ for different values of σ. We set $\mu = 0.02$ and $\eta = 8$. Observe that the performance gap between the proposed distributed algorithm and the centralized solution increases as the error vector $\boldsymbol{w}_\star^\delta$ increases.

Finally, we illustrate in Fig. 5 *(right)* the influence of μ and η on the performance of the proposed algorithm (17). For comparison purposes, we set $\sigma = 1$ for all the μ and η. We observe that, as expected, the larger μ is, the faster the convergence rate is but the worse the MSD performance is. In addition, the performance improves by increasing η.

V. Conclusion

In this work, we proposed a distributed multitask LMS algorithm for solving problems that require the simultaneous estimation of multiple parameter vectors that are related locally via linear equality constraints. We approximate the

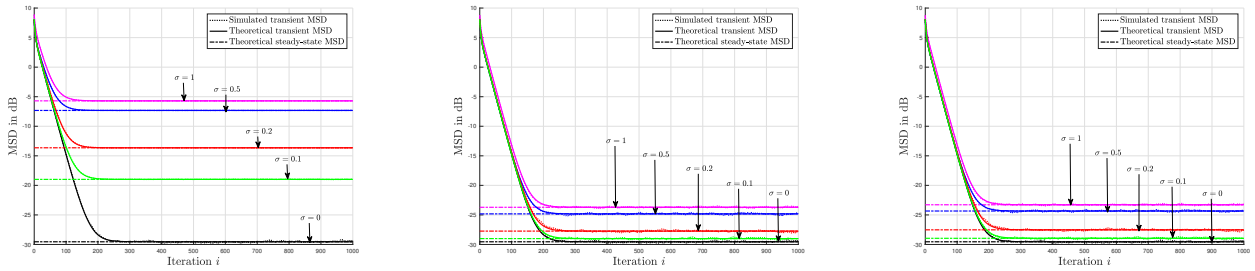

Fig. 4: MSD with respect to $\boldsymbol{w}^o$(left), $\boldsymbol{w}^o(\eta)$ (*middle*) and $\boldsymbol{w}^\star$ (*right*) for different σ.

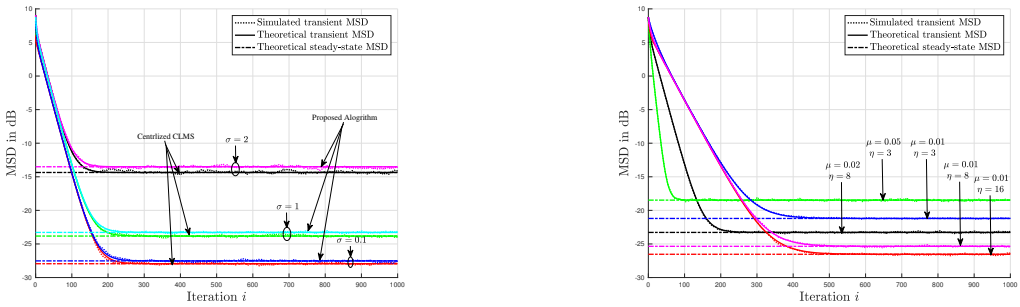

Fig. 5: MSD comparison for the imperfect model scenario.

original constrained problem by an unconstrained one based on the penalty method. The behavior of the algorithm in the mean and in the mean-square-error sense was analyzed. Finally, the simulations showed the efficiency of the proposed method.

REFERENCES

[1] A. H. Sayed, *Diffusion adaptation over networks*. Academic Press Library in Signal Processing, 2013, vol. 3.

[2] A. H. Sayed *et al.*, "Adaptation, learning, and optimization over networks," *Foundations and Trends in Machine Learning*, vol. 7, no. 4-5, pp. 311–801, 2014.

[3] A. H. Sayed, "Adaptive networks," *Proceedings of the IEEE*, vol. 102, no. 4, pp. 460–497, 2014.

[4] Z. J. Towfic and A. H. Sayed, "Adaptive penalty-based distributed stochastic convex optimization," *IEEE Transactions on Signal Processing*, vol. 62, no. 15, pp. 3924–3938, 2014.

[5] J. Chen and A. H. Sayed, "Distributed pareto optimization via diffusion strategies," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, no. 2, pp. 205–220, 2013.

[6] J. Chen, C. Richard, and A. H. Sayed, "Multitask diffusion adaptation over networks," *IEEE Transactions on Signal Processing*, vol. 62, no. 16, pp. 4129–4144, 2014.

[7] ——, "Diffusion LMS over multitask networks," *IEEE Transactions on Signal Processing*, vol. 63, no. 11, pp. 2733–2748, 2015.

[8] R. Nassif, C. Richard, A. Ferrari, and A. H. Sayed, "Multitask diffusion adaptation over asynchronous networks," *IEEE Transactions on Signal Processing*, vol. 64, no. 11, pp. 2835–2850, 2016.

[9] ——, "Proximal multitask learning over networks with sparsity-inducing coregularization," *IEEE Transactions on Signal Processing*, vol. 64, no. 23, pp. 6329–6344, 2016.

[10] N. Bogdanovic, J. Plata-Chaves, and K. Berberidis, "Distributed incremental-based LMS for node-specific adaptive parameter estimation," *IEEE Transactions on Signal Processing*, vol. 62, no. 20, pp. 5382–5397, 2014.

[11] J. Plata-Chaves, N. Bogdanović, and K. Berberidis, "Distributed diffusion-based LMS for node-specific adaptive parameter estimation," *IEEE Transactions on Signal Processing*, vol. 63, no. 13, pp. 3448–3460, 2015.

[12] J. Plata-Chaves, M. H. Bahari, M. Moonen, and A. Bertrand, "Unsupervised diffusion-based LMS for node-specific parameter estimation over wireless sensor networks," in *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*. IEEE, 2016, pp. 4159–4163.

[13] J. Chen, C. Richard, A. O. Hero, and A. H. Sayed, "Diffusion LMS for multitask problems with overlapping hypothesis subspaces," in *IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, 2014.

[14] J. Chen, C. Richard, and A. Sayed, "Multitask diffusion adaptation over networks with common latent representations," *IEEE Journal of Selected Topics in Signal Processing*, 2017.

[15] C.-K. Yu and A. H. Sayed, "Learning by networked agents under partial information," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 3874–3878.

[16] D. P. Bertsekas, *Network optimization: continuous and discrete models*. Athena Scientific, 1998.

[17] C. Shen, T.-H. Chang, K.-Y. Wang, Z. Qiu, and C.-Y. Chi, "Distributed robust multicell coordinated beamforming with imperfect csi: An ADMM approach," *IEEE Transactions on signal processing*, vol. 60, no. 6, pp. 2988–3003, 2012.

[18] J. F. Mota, J. M. Xavier, P. M. Aguiar, and M. Puschel, "Distributed basis pursuit," *IEEE Transactions on Signal Processing*, vol. 60, no. 4, pp. 1942–1956, 2012.

[19] R. Nassif, C. Richard, A. Ferrari, and A. H. Sayed, "Diffusion LMS for multitask problems with local linear equality constraints," *arXiv:1610.02943*, 2016.

[20] B. T. Polyak, *Introduction to Optimization*. Optimization Software, New York, 1987.

[21] M. S. Bazaraa, H. D. Sherali, and C. M. Shetty, *Nonlinear programming: theory and algorithms*. John Wiley & Sons, 2013.

[22] A. H. Sayed, *Adaptive filters*. John Wiley & Sons, 2011.

[23] O. L. Frost, "An algorithm for linearly constrained adaptive array processing," *Proceedings of the IEEE*, vol. 60, no. 8, pp. 926–935, 1972.